How to connect design thinking and cyber-physical systems: the s*IoT conceptual modelling approach

Michael Walch
University of Vienna
michael.walch@univie.ac.at

Dimitris Karagiannis
University of Vienna
dk@dke.univie.ac.at

Abstract

The alignment of enterprise models and information systems is a factor that influences the efficiency of enterprise practices. Considering the changing landscape in the age of the fourth industrial revolution, it is imperative that alignment methodologies are evolved with the progression of enterprise models and the transformation from information systems to cyber-physical systems (CPSs). This issue was dissected in three layers – scenario layer, modelling layer, and run-time environment. In this structure, design thinking and CPSs were extended from the scenario layer and the run-time environment to the modelling layer. Focusing on the modelling layer, progress was made towards composing "smart" models that innovate enterprise models according to novel influences from design thinking while abstracting from run-time environments that CPS provide. The hypothesis was to consider the automated transformation of knowledge as an axle around which artifacts on the modelling layer revolve. Based on this hypothesis, the modelling layer was structured in a modelling hierarchy, in which a metamodel was defined using a metamodelling platform. The metamodel is the direct model of modelling methods which were used to build "smart" models that connect design thinking and CPSs.

1. Introduction

The digital transformation age is revolutionizing the co-dependence of society and technology, as a composition of new conceptual designs, modelling artifacts, and socio-technical systems emerges by embracing digital innovation for just about everything [1, 2]. In a business context, enterprise models[1] and the means to build them lie at the heart of this revolution. They are the artifacts that realize conceptual designs while being put to use by socio-technical systems. As the digital transformation age brings about a revolution of conceptual designs and socio-technical systems, enterprise models have to adopt accordingly.

The transformation of innovation into business value is not a straight forward task but rather a wicked problem [4]. Design thinking is one approach to tackle this problem by conjuring and capturing conceptual designs in tangible artifacts, some of which – as illustrated in this paper – can be refined into enterprise models by means of conceptual modelling. Standardized enterprise models are often not sufficient for this task. Rather, agile modelling method engineering can be applied to progress enterprise models in an innovative direction [5].

The development of enterprise models is one task in a business context. Another one is the operationalization of enterprise models, which implies that all the knowledge that is recorded in enterprise models for human interpretation has to be put to use. For quite some time now, information systems have supported humans in the task of putting enterprise models to use. To guarantee smooth enterprise operations, enterprise models have to be appropriately aligned with suitable information systems.

In the digital transformation age, the task of putting enterprise models to use is shifted from humans towards automated tasks performed by machines and in particular by cyber-physical systems (CPSs) [6, 7]. The complexity of this progress stems from two challenges of CPSs: their high variability at design-time and their complex dynamics at run-time. High variability at design-time is a result of applying design thinking & conceptual modelling to hypothetical application scenarios of CPSs, while complex dynamics at run-time results from the reality of executed CPS behaviour.

While design thinking, conceptual modelling, and agile modelling method engineering can provide means to record conceptual designs in progressive enterprise models for human interpretation, the question is how to put these models to use by executing them in

[1] The term *enterprise models* is used in a broad sense that includes business models on different organizational layers [3].

URI: https://hdl.handle.net/10125/60161
ISBN: 978-0-9981331-2-6

an automated manner. That is because currently few enterprise models can be executed and of those far fewer are rewarding to use [8]. Consequently, research is required on how to reduce the human effort when enterprise models are put into operation. In particular, this paper asks the question how progressive enterprise models can be aligned with CPSs, considering that such an alignment should enable the automated execution of said models. The problem is to connect the decomposition of conceptual designs for business innovation and for hypothetical application scenarios of CPSs with the abstraction of executed CPS behaviour. Furthermore, eliminating the dependence on human interpretation and action when enterprise models are put to use requires that the intelligence required to turn humanized knowledge into machine interpretable form has to be reproduced either by implementing it inherent to CPSs or by extending progressive enterprise models even further. The second option is the one favoured in this paper which composes the necessary methodological framework in Section 2. With respect to this framework, the state of the art is discussed in Section 3. Section 4 applies the methodological framework in a case study which involves the use of a metamodelling platform to build the metamodel that is necessary for connectivity between design thinking and CPSs. The metamodel from Section 4 is the direct model of modelling methods which are applied as tools in laboratory experiments for validation purposes in Section 5. Section 6 concludes the topics.

2. The s*IoT Methodology

This section provides a methodology to conduct applied research on the introduced topic. Based on a scientific approach [9], the methodology is presented by addressing its goals, conceptual framework, contributions, methods, and validation strategies. An instance of the methodology is applied to conduct the case study that is covered in Section 4 and 5.

2.1. Goals

The overall goal of the s*IoT methodology is to facilitate connectivity between design thinking and CPSs to enable synergistic effects in the digital transformation age. By considering enterprise models as first-class citizens, the goal can be broken down into two subgoals. The first is to innovate and record conceptual designs from the business domain in progressive enterprise models, which are a subset of conceptual models. The second is to align the resulting models and CPSs. Together, the subgoals require the search for mutual artifacts (as seen in Fig. 1).

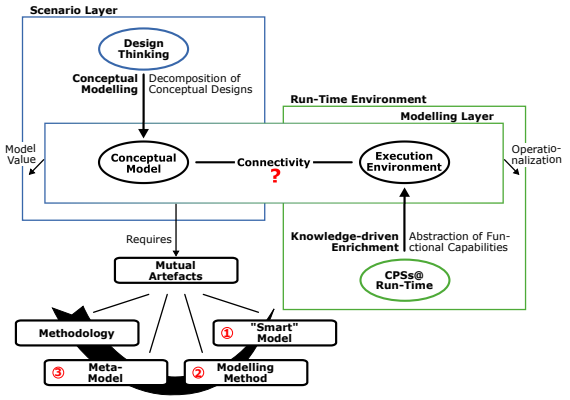

Figure 1. Connectivity between design thinking and CPSs relies on mutual artifacts on the modelling layer.

The immediate artifacts under scrutiny are progressive enterprise models that contain the intelligence and abilities required for their operationalization. The term ”smart” models has been coined for these kinds of models. Modelling methods frame the search for models in general, which is true for ”smart” models as well. As the modelling methods for ”smart” models do not yet exist, this methodology aims to search for them. Metamodels are the direct models of modelling methods and frame the search. Consequently, the goal of the s*IoT methodology is to search for (1) ”smart” models that are co-constructed by design thinking and CPSs, (2) modelling methods for building ”smart” models, and (3) a model of the modelling method that covers invariants of ”smart” models on the meta-level, i.e., a metamodel. The desired effect of these artifacts is to increase the efficiency of enterprise practices.

On a methodological level, the first subgoal is covered by the practiced reality of design thinking, conceptual modelling, and agile modelling method engineering, which is why it can be excluded for the most part from a discussion in this section. The second subgoal, however, requires further examination. The relation between models and CPSs is that different domain-specific modelling methods are applied to understand, plan, and operate CPSs,[2] which results in heterogeneous models [11, 12, 13]. The resulting models separate into two general types based on established community practices: models that decompose conceptual designs to tackle the high variability of CPS at design-time and models that abstract functional capabilities to tackle the complex dynamics of CPS at run-time. For the clarity of presentation, these two types of models are addressed as conceptual models and execution

[2]Additional benefits of a model-based approach can be found in [10].

environments respectively. Conceptual models are symbolic representations of a problem domain, made at design-time by humans for human use [14]; and execution environments are practical reflections of computation, networking, and physical processes, that provide a context for operationalization at run-time [15]. Correspondingly, conceptual models are built in conceptual modelling to represent design-time aspects of CPSs [16]; and execution environments are built using knowledge-driven enrichment to reflect run-time aspects of CPS [17]. A qualitative distinction between the two types of models results from their different purpose, which is reinforced by established community practices. The problem is that visionary opportunities could emerge from synergies between conceptual models and execution environments in the digital transformation age. For example, order picking in the factory of the future could benefit from connectivity between an innovative process model that decomposes progressive concepts for this scenario from design thinking and an execution environment like a microservice portal that abstracts functional capabilities from an available pick-and-place machine, resulting in a potentially more agile, transparent, and effective automation. Consequently, it is no longer justified to isolate conceptual designs and functional capabilities in different types of models. Rather, established communities have to bridge existing gaps to work towards the hypothesis that connectivity between design thinking and CPSs is required in terms of design-time and run-time aspects; and in terms of business concepts and the behaviour, structure, and function of CPSs.

2.2. Conceptual Framework

Integrated with computer science, design science is researching designs of information systems in form of purposeful artifacts for information systems engineering [19, 20]. Likewise, a specialization of the design science paradigm with a model-based approach is searching for artifacts on a modelling layer (as seen in Fig. 2). Applied to the introduced issue of integrating digital technologies into just about everything, new

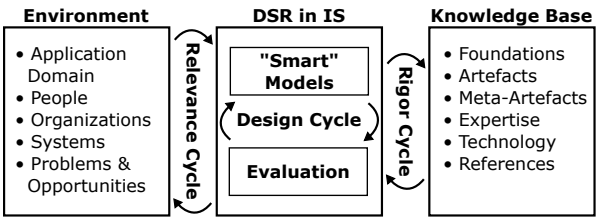

Figure 2. Agility within s*IoT. Adopted from [18].

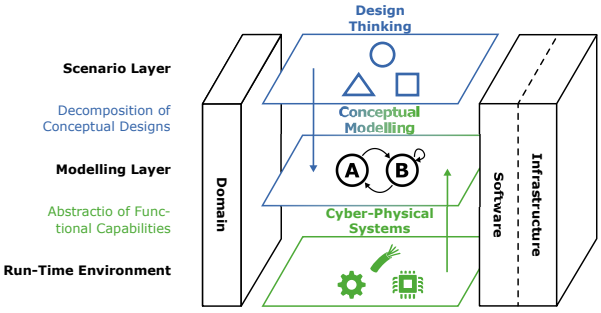

Figure 3. The three layer s*IoT architecture.

and evolved model-based artifacts are required for connectivity between design thinking and CPSs. The instantiation of the design science paradigm for the introduced issue results in a three layer architecture (as seen in Fig. 3) which contains the scenario layer, the modelling layer, and the run-time environment. The conceptual framework for building model-based artifacts is based on this three layer architecture.

The scenario layer covers conceptual designs that are – by design thinking – transformed from mental models into tangible artifacts. When design thinking is applied in combination with conceptual modelling in a business context, a transformation from the scenario layer to the modelling layer decomposes conceptual designs in enterprise models, which provides model value for human stakeholders [21]. The necessary enterprise models can be evolved by agile modelling method engineering. To put enterprise models to use, a transformation into an operation environment is required. When CPSs are considered, the operation environment structures itself into two categories: run-time environment and execution environment. The former frames the realization of CPSs and their behaviour into the non-deterministic physics of reality, of which the latter is a formal abstraction for machine interpretation that can be thought of conceptually. When relating enterprise models and operation environments, two methodological directions from systems modelling are relevant. The first is model-driven engineering, which continues the decomposition of model-based conceptualizations to deploy CPSs. Thereby, CPS designs are derived in relation to enterprise models at design-time, which results in conceptual models of CPS that are infused with (semi)-formal semantics to enable the (semi)-automatic deployment of run-time environments. The second direction, which abstracts model-based conceptualization from CPS ecosystems, is lacking a decisive term. Nevertheless, the idea is that execution environments, which provide means for operationalization on the modelling layer, are linked to CPSs at run-time. The nature of this link is that of a

feedback loop between abstract categories in execution environments and their concrete physical realization in run-time environments. One example for execution environments is found in service oriented architectures that define aggregated services for applications, based on computation, networking, and physical processes that some run-time environment offers. The second direction is the one this methodology focuses on, as the feedback loop enables a more agile composition of enterprise models, conceptual models of CPSs, and their operation environment.

In ”smart” models, the function and structure of the composition and its resulting behaviour is enabled by artificial intelligence technologies. To design & engineer ”smart” models, a meta-level view considers the purposeful artifacts that result from design science as systems under study by design science. Consequently, the modelling layer is structured in a modelling hierarchy. A transformation between artifacts on different levels of the modelling hierarchy is possible using metamodel-based implementation, e.g., metamodels are direct models of modelling methods, which in turn are used to build models [22].

2.3. Contributions

Possible contributions provide a point of departure for research that plans to apply the s*IoT methodology. Thereby, connectivity between design thinking and CPSs is broken down into smaller issues worth solving. This subsection structures possible research questions in three overall categories: (1) which research topics benefit from the methodology, (2) what is the nature of connectivity, and (3) how to facilitate research, education, and practice in a community.

The first category examines topics that can be tackled by applying the s*IoT methodology. The common theme of these topics is that a combination of business innovation and technological innovation provides immediate benefits but relies on knowledge engineering for connectivity on the modelling layer.

- Knowledge Creation by Artificial Recognition – possible scenarios revolve around CPSs that recognize the real world by coordinating computer boards, IoT protocols, and sensors, which is enabled through knowledge engineering, e.g., by artificial neural networks, hidden Markov models, and support vector machines.
- Knowledge Execution through Reasoning & Planning – possible scenarios revolve around CPSs that provide innovative services by controlling kinematics of actuators in difficult environments, which is enabled through knowledge engineering, e.g., by ontologies, rule engines, and fuzzy logic.
- Knowledge Transfer with (Eco-)System Architectures – possible scenarios revolve around communication, negotiation, and collaboration between humans and heterogeneous CPSs, which is enabled through knowledge engineering, e.g., by semantic alignment, microservice frameworks, and multi-agent systems.
- Knowledge Validity & Traceability – possible scenarios revolve around humanized white-box models that explain the behaviour or CPSs, which is enabled through knowledge engineering, e.g., by decision trees, evolutionary computation to invert neural networks, and data visualization.

The second category tries to understand the nature of connectivity between design thinking and CPSs. By applying the s*IoT methodology to co-construct collective artifacts, the issue boils down to a composition of (1) design thinking & conceptual modelling, which transforms conceptual designs for innovative business scenarios and design-time aspects of CPSs into conceptual models, and (2) conceptual modelling & knowledge-driven enrichment, which transforms atomic and also abstract, intricate functional capabilities of CPS run-time environments into execution environments. Fig. 4 condenses the resulting problem of aligning conceptual models and execution environments in ”smart” models. A closer look unveils research questions regarding the nature of connectivity. In detail, connectivity is required between innovative business scenarios, CPS designs from which requirements for CPS capabilities can be inferred, functional capabilities of CPSs, and their realization. However, alignment is facilitated on the modelling layer in a modelling hierarchy. The expected outcome of this are metamodels, modelling methods, and ”smart” models. To design & engineer artifacts in the modelling hierarchy, the following questions have to be answered.

- What is the common ontological structure for models of conceptual designs and functional

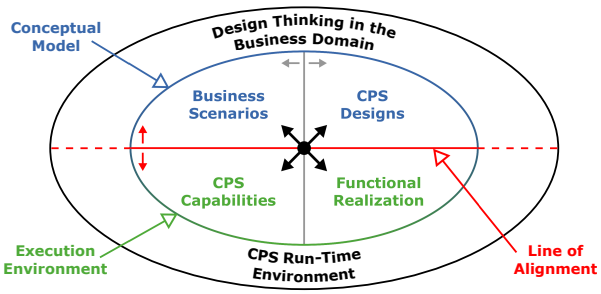

Figure 4. The four quadrants for alignment.

capabilities? How is it realized in concrete metamodels and modelling methods?

- How are models that use concepts and capabilities realized by ”smart” models? How efficient are the modelling methods at building ”smart” models?
- What tooling has to be provided to construct metamodels, modelling methods, and ”smart” models? How can it be improved and distributed?
- Which aspects of conceptual models and execution environments should be connected in detail? How is connectivity realized, e.g., is it done on-line or off-line, synchronous or asynchronous, exogenous or endogenous?
- How can connectivity benefit from artificial intelligence technologies, knowledge representation schemes, and semantic services?

The third category is concerned with the interdisciplinary competence and mentality between scientific and professional communities. The separation between conceptual models and execution environments is not only found in artifacts, but also in established community practices. The s*IoT methodology is interested in enabling different communities from different domains to cooperate. Cooperation is required to contribute in a business context to a common understanding of design thinking and CPSs. The importance of this pragmatic aspect cannot be underrated if the existing communities are to prevail and thrive in the digital transformation age.

2.4. The Decomposition and Abstraction Methods

To build ”smart” models in the three layer architecture that has been introduced as part of the conceptual framework, it is necessary to extend the scenario layer to the modelling layer, to extend run-time environments to the modelling layer, and to connect both in mutual artifacts. Similarly, a modelling method for combining the decomposition of conceptual designs and the abstraction of functional capabilities is necessary. The ingredients of this combined modelling method are design thinking, conceptual modelling, and knowledge-driven enrichment (as seen in Fig. 5).

Design thinking & conceptual modelling starts as a process that is assigned to the scenario layer: creative and educated minds conceive mental models of problem domains and collaborate on them. These mental models are then decomposed at design time by the formalization of conceptual designs in conceptual models, which extends the scenario layer to the modelling layer. Conceptual models are structured by *models of concepts* and *models that use concepts* [23]. The former represent invariants of problem domains, which is a prerequisite for finding more specific artifacts of the latter type.

Conceptual modelling & knowledge-driven enrichment starts as a process that is assigned to the run-time environment: the concrete physical characteristics and configurations of CPSs provide a context for operationalization. The means necessary for operationalization are abstracted by the composition of functional capabilities in execution environments, which extends the run-time environment to the modelling layer. Execution environments determine *models of functional capabilities* which provide a run-time environment for *models that use functional capabilities*. As a consequence, specific functional capabilities make up more intricate, abstract ones. The specific end of the spectrum more directly relates to the run-time environment that CPSs provide and grounds the model hierarchy in reality. Likewise, execution environments abstract the automated processing of data, information, and formal knowledge, by processors, networks, operating systems, and so on. Thereby, a loss of detail is desired to gain usability when modelling functional capabilities.

However, the results of decomposition and abstractions would be two types of models – conceptual models and execution environments – which would contain knowledge in different form (humanized and machine-executable). Instead, ”smart” models require a transformation of knowledge on the modelling layer to bridge the gap. To transform between informal knowledge (which is found predominantly in conceptual models) and formal knowledge, information, and data (which is predominantly found in execution environments), artificial intelligence technologies have to be embedded in modelling methods that combine the decomposition and abstractions methods, in addition to knowledge engineering methods.

To enable the combination of decomposition, abstraction, and knowledge engineering in a single modelling method, metamodelling has to be considered, as it provides the means to engineer (hybrid) modelling methods [24]. As mentioned, metamodels are direct models of modelling methods, which in turn are deployed as tools to build models [22]. The process that leads to these artifacts is practiced in five phases [25]: ”Create”, in which modelling requirements are conceived; ”Design”, in which the ontological scope of modelling requirements is constructed; ”Formalize”, in which the results from the previous phase are represented in a non-ambiguous and formal way by metamodels; ”Develop”, in which the ontological scope is combined with an operational scope in modelling

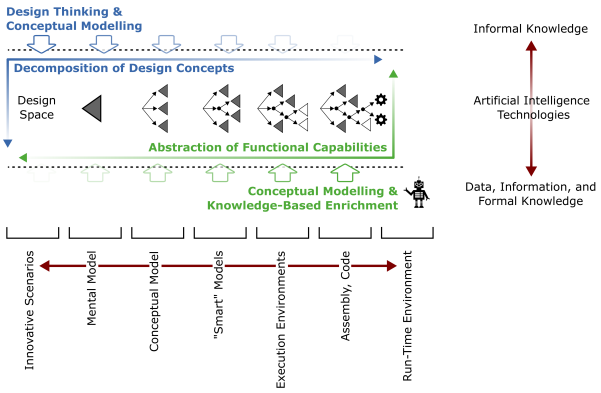

Figure 5. Decomposition and abstraction in a design context

methods; and ”Deploy”, in which a modelling tool is evaluated for upcoming iterations. For increased agility, the design & engineering lifecycle can be extended with additional feedback channels [25].

2.5. Validation Strategies

To validate artifacts that result from applying the s*IoT methodology, naturalistic ex post evaluation [26] is proposed. Thereby, connectivity between design thinking and CPSs – through the composition of conceptual models and execution environments – is conducted in experiments. To conduct the experiments, a laboratory has to be set up that realizes a space for the scenario layer, modelling layer, and run-time environment – to facilitate the conceptualization of scenarios from the digital transformation age, the implementation of mutual artifacts that fuse & evolve conceptual models and execution environments, and the deployment of CPSs in run-time environments that provide means for operationalization. Focusing on the modelling layer, the experiments should revolve around ”smart” models. Thereby, experimental evaluation is obtained for the applicability of a metamodel when constructing modelling methods (that are deployed as tools) in the task of modelling method engineering; and for the applicability of modelling methods and tools when building ”smart” models in the task of domain-specific modelling.

3. Related Work

A body of work exists on standardized, but also on domain-specific enterprise models that record innovative scenarios, mental models, and conceptual designs [16]. A body of work also exists on execution environments that control CPSs and give an account of their behaviour. Such work and a combination thereof is situated on a model level in the modelling hierarchy and cannot be presented adequately by this paper. Rather, this section provides an overview of work done on the meta-level.

Two tracks can be identified on the meta-level. In the first track, metamodels are discussed for run-time environments that contain models. In the second, metamodels are discussed for models that are executed. For the first track, prominent examples can be found for the topic of digital service discovery, where WSDL provides a web service description [27], Hypercat and its RDF-based alternatives provide semantic service catalogs [28], and XMPP provides means for agents to detect their presence [29]. These examples are standards for models that are provided by run-time environments for consumes that want to use services. These standards define languages, procedures, and are foundations for mechanisms & algorithms. Therefore, they operate on the meta-level. In the second track, models are extended by or transformed into execution environments. These models can be standard model types like UML which are extended by fUML [30] for an executable subset of UML or executed via code generation in model driven engineering [31]. These models can also be domain-specific model types like SysML which require dedicated execution environments [32].

The open issue is to find a holistic structure in which the existing metamodels can be integrated as metamodel building blocks, while tackling the introduced topic of connecting design thinking and CPSs. The issue is relevant as it is not feasible to connect execution environments and conceptual models on the meta-level, yet alone on the model-level, for each type of progressive enterprise model and execution environment. Rather, a holistic metamodel for the issue at hand can be used to structure, integrate, and reuse different overlapping approaches on the matter.

4. Case Study Setup

The s*IoT methodology could be used in a wide variety of research projects. This claim has been verified by a concrete collaboration between previously unrelated research projects. Using the introduced methodology, a correlation between the projects could be found, which enabled deeper collaboration [33]. In this section, an instance of the methodology is applied by focusing on a single research question, which is to find a metamodel for connectivity between design thinking and CPSs.

In particular, the idea is to model and bridge the capability gap between (1) capability requirements for CPSs that can be inferred from enterprise models at

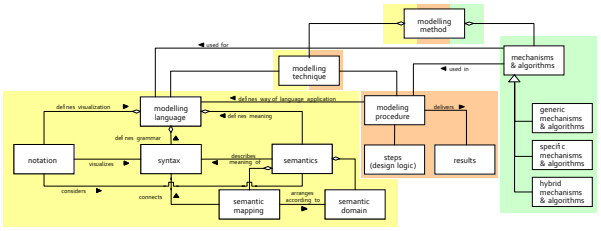

Figure 6. Framework for modelling method engineering [22].

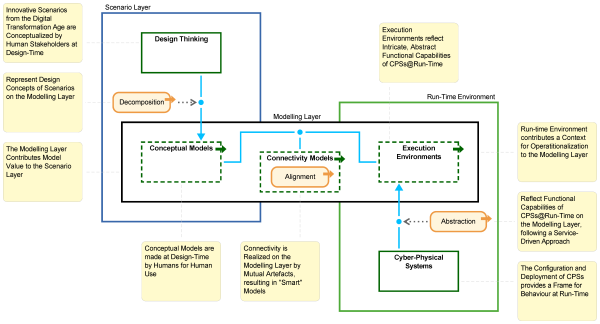

Figure 7. This level (0) of the metamodel shows the abstract concepts that are necessary for connectivity between design thinking and CPSs.

design-time and (2) functional capabilities that reflect the operation of CPSs at run-time. As a consequence, capability requirements and functional capabilities co-construct the axle around which enterprise models and execution environments revolve, which involves a transformation of knowledge using artificial intelligence technologies. The metaphorical axle has to be realized on the modelling layer in the modelling hierarchy.

To build ”smart” models that feature the properties of the previous paragraph, a dedicated modelling method is required. To build modelling methods, a modelling method engineering framework has been proposed (as seen in Fig. 6). The framework relies on a metamodel that is realized by a modelling language, modelling procedures, and mechanisms & algorithms.

The direct model of a modelling method is a metamodel. Metamodelling platforms implement a meta^{2}model and enable the construction of metamodels. Some freely available choices for metamodelling platforms are ADOxx (www.adoxx.org), MetaEdit+ (www.metacase.com/mep), and EMF (www.eclipse.org/emf). However, when selecting a specific platform, one has to take into account how components of the platform support design & engineering [34]. Alternatively, a domain-specific metamodelling language (MM-DSL) could be used for cross-platform support [18]. After the metamodel is defined in a metamodelling platform, it can be turned into a modelling method that includes tool support. Consequently, modelling methods for the problem at hand can be derived from the metamodel that is presented in this section. Furthermore, the resulting modelling methods can be applied as a tool in domain-specific modelling to build ”smart” models.

In Fig. 7-9, the metamodel of ”smart” models is described in different levels of detail. The starting point is the three layer architecture as discussed. Details are added on each level until a full fledged metamodel emerges that makes use of the elements in the discussed modelling method engineering framework – modelling language, modelling procedures, and mechanisms & algorithms.

By adding details to the metamodel, the nature of connectivity becomes more clear. Likewise, the extension of design thinking and CPSs to the modelling layer becomes more clear, up to the point where it can be supported by automation. The same is true for connectivity between conceptual models and execution environments. For example, conceptual models can be further decomposed into hybrid models that are a composition of domain models, enterprise models, and scenario models; while execution environments further abstract CPSs into configuration models, capability models, and CPS class models. Connectivity then boils down to multi-agent models that instantiate CPS class models and negotiate between the functional capabilities they offer and the requirements for capabilities from hybrid models, which can be done using artificial intelligence technologies.

A more detailed version of the metamodel is not suitable for presentation within this paper due to the size of the illustration (it can be accessed at www.omilab.org/sIoT). At some point, details that are specific to a metamodelling platform have to be added. This process is shown in the validation section, where the ADOxx metamodelling platform is used to specialize the metamodel. The results are modelling methods and tools that enable experiments that allow us to show how design thinking and CPSs can be connected conceptually and in practice.

5. Case Study Validation within OMiLAB

The previous section set up a metamodel of ”smart” models, which is the foundation for the case study that is discussed in this section. In the case study, the validation strategy of the s*IoT methodology is instantiated and applied to validate the metamodel, which requires to conducting experiments in a laboratory. Thereby, naturalistic ex post validation is looking at how

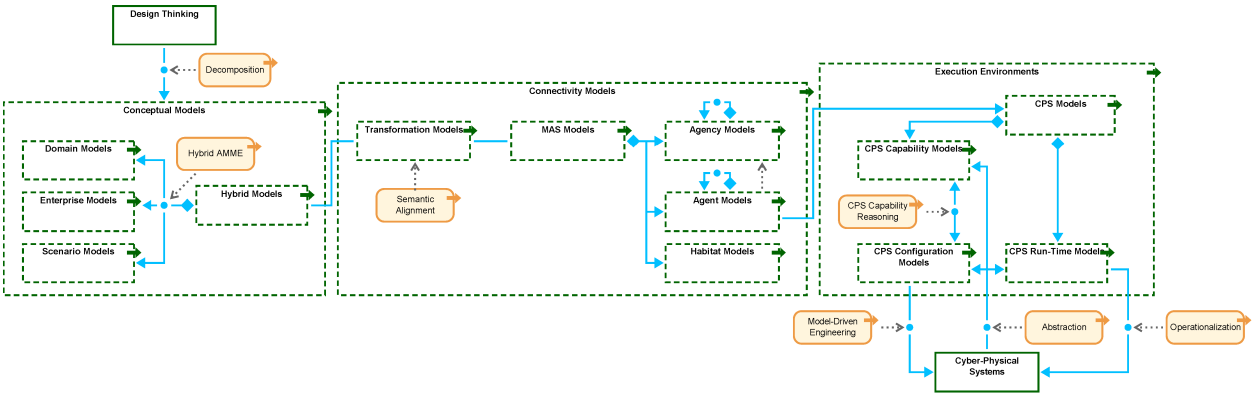

Figure 8. This level (1) of the metamodel shows the concepts that connect conceptual models and execution environments.

artifacts of a research progress perform. Likewise, the metamodel of ”smart” models and implicitly also the s*IoT methodology are validated by the perceived success when corresponding modelling methods and tools are used to conduct experiments with ”smart” models. Both, laboratory and experiments, are described in this section.

The laboratory for this study is provided as part of the OMiLAB non-profit organization. The organization is a global network of local research, application, and education spaces. The individual spaces are realized by different collaborators like research groups and industry partners, each focusing on a topic surrounding metamodelling. OMiLAB Vienna hosts different

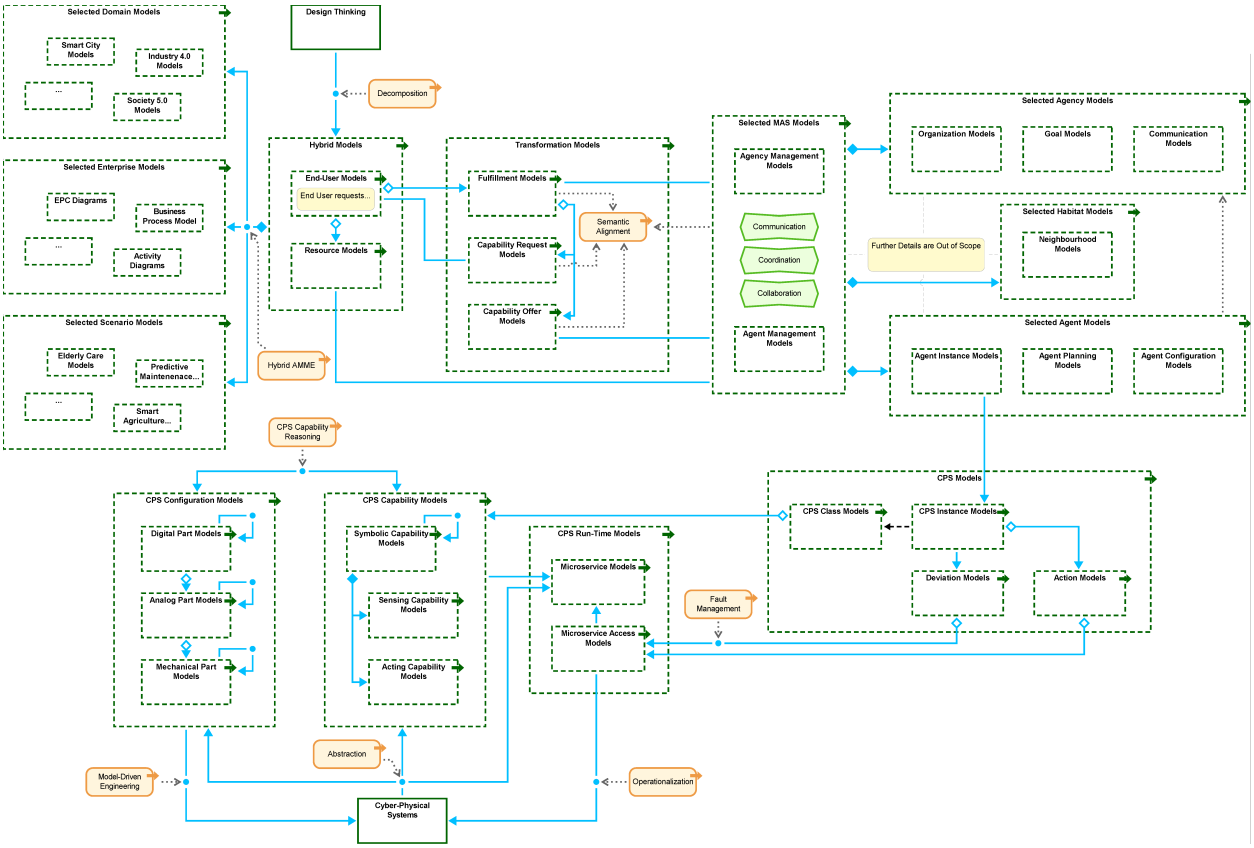

Figure 9. This level (2) and more detailed levels of the metamodel can be enriched with details specific to metamodelling platforms.

Table 1. Summary from 40 experiments.

	Moveable CPSs			Stationary CPSs		
	Drive	Fly	Walk	Arms	Sense	ARS
Industry 4.0	2 EMO		2 EGI	6 EMO	1 EIO	
Recreation	3 ESR	1 ENO	1 ES	4 ECGO	3 EI	1 ITV
Auton. Driving	11 EIFN				1 E	1 IT
Smart Assistant	1 IN		1 ES	5 EGO		

Constraint Satisfaction (C), Fuzzy Logic (F), Genetic Algorithms (G), Image Recognition (I), Multi-Agent System (M), Neural Network (N), Object Tracking (T), Ontology (O), Rule Engine (E), Speech Recognition (R), Speech Synthesis (S), Virtual Reality (V)

spaces, one of them being the Digital Product Space. One part of this space is the OMiRob laboratory, which is a physical and virtual realization of the introduced three layer architecture. The scenarios layer is realized by a socio-technical system that decomposes conceptual designs semi-automatically into conceptual models. The modelling layer is built around the ADOxx community and embraces the use of the ADOxx metamodelling platform for building modelling methods. The run-time environment consists of CPSs in form of different types of robots of varying complexity.

Within the OMiRob laboratory, experiments were conducted (some of which are detailed on http://austria.omilab.org/psm/omirob). The invariant in these experiments was to build a modelling method that covers a building block of the introduced metamodel. Thereby, an important aspect was to move the source of intelligent behaviour away from the run-time environment, where minimal, atomic, and ”assembler-like” capabilities were sufficient, towards ”smart” models. However, the task was not to implement instances of intelligent behaviour on the model level, but to provide it as part of a modelling method that a metamodel building block is the direct model of. The idea is to collect a repository of metamodelling building blocks that modelling method engineers can use as they seem fit for building domain-specific modelling methods for ”smart” models that can be put to use in an automated manner.

The experiments themselves are based on innovative scenarios in the digital transformation age and the run-time environment that is provided by OMiRob. An overview is provided in Table 1, which groups the experiments in categories for scenarios and CPSs. It also list artificial intelligence technologies that were used. Examples are the use of robotic arms and self driving cars for a delivery on demand scenario, the use of humanoid robots as smart assistants in a health care scenario, or the use of quadcopters in a recreational tourism scenario. The modelling methods that were developed to conduct the experiments decomposed conceptual designs, abstracted functional capabilities, and employ artificial intelligence technologies in ”smart” models. To do so, they refined a part of the introduced metamodel with additional detail in terms of modelling language, modelling procedures, and mechanisms & algorithms until it was possible to deploy a modelling tool using ADOxx. The successful application of this modelling tool verifies the corresponding metamodel building block. In the same manner, the s*IoT methodology is verified as well.

6. Conclusion

Overall, this study provides general and systematic theories by discussing the s*IoT methodology, a metamodel of ”smart” models, and proof-of-concept modelling methods that compose and refine metamodel building blocks in tools that individuals can use as they seem fit to understand and influence the co-dependence of society and technology by building ”smart” models that connect design thinking and CPSs. Connectivity was achieved on the modelling layer in laboratory experiments by utilizing decomposition, abstraction, and artificial intelligence technologies to compose progressive enterprise models, which extend design thinking, and execution environments, which extend run-time environments of CPSs. The insights that were gained from the experiments suggest that the reemerging potential of machine learning and formal mathematical methods could provide yet to be explored benefits on the modelling layer for the issue at hand. Future work aims to develop a modelling method not only for specific metamodel building blocks, but for the integrated metamodel of ”smart” models. The result will be called the s*IoT modelling method. The anticipated benefit is that this modelling method could be used in a wide variety of experiments and real world applications by people unfamiliar with modelling method engineering.

References

[1] J. Gray and B. Rumpe, “Models for the digital transformation,” *Software & Systems Modeling*, vol. 16, no. 2, pp. 307–308, 2017.

[2] K. Smolander, M. Rossi, and S. Pekkola, “Infrastructures, integration and architecting during and after digital transformation,” in *Proceedings of JSOS’17*, (Buenos Aires, Argentina), pp. 23–26, 2017.

[3] H. Österle and D. Blessing, “Ansätze des Business Engineering,” *HMD*, vol. 42, no. 241, pp. 7–17, 2005.

[4] T. Kelly, *The Art of Innovation: Lessons in Creativity from IDEO, America’s Leading Design Firm*. New York: Broadway Business, 2001.

[5] D. Karagiannis, "Agile modeling method engineering," in *Proceedings of PCI'15*, (New York, NY, USA), pp. 5–10, 2015.

[6] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 3, pp. 18 – 23, 2015.

[7] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[8] B. Thalheim, "Conceptual model notions - a matter of controversy: Conceptual modelling and its lacunas," in *Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr* (J. Michael, C. Steinberger, V. A. Shekhovtsov, S. Ranasinghe, and F. A. Machot, eds.), pp. 9–27, Hagen: EMISA, 2018.

[9] J. A. Maxwell, *Qualitative Research Design: An Interactive Approach*, vol. 41 of *Applied Social Research Methods*. London: Sage Publications, 2012.

[10] B. Selic, "The pragmatics of model-driven development," *IEEE Softw.*, vol. 20, no. 5, pp. 19–25, 2003.

[11] A. R. da Silva, "Model-driven engineering: A survey supported by the unified conceptual model," *Computer Languages, Systems & Structures*, vol. 43, pp. 139 – 155, 2015.

[12] C. Schlegel, T. Haßler, A. Lotz, and A. Steck, "Robotic software systems: From code-driven to model-driven designs," in *Proceedings of ICAR'09*, (Munich, Germany), pp. 1–8, 2009.

[13] B.-H. Schlingloff, "Cyber-physical systems engineering," in *Engineering Trustworthy Software Systems: First International School, SETSS 2014, Tutorial Lectures* (Z. Liu and Z. Zhang, eds.), pp. 256–289, Cham: Springer, 2016.

[14] J. Mylopoulos, "Conceptual modelling and Telos," *Conceptual Modelling, Databases, and CASE: an Integrated View of Information System Development*, pp. 49–68, 1992.

[15] A. Burns and A. Wellings, *Real-Time Systems and Programming Languages: Ada, Real-Time Java and C/Real-Time POSIX (4th Edition)*. Longmain: Addison Wesley, 2005.

[16] D. Karagiannis, H. C. Mayr, and J. Mylopoulos, *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*. Cham: Springer, 2016.

[17] M. Walch, "Knowledge-driven enrichment of cyber-physical systems for industrial applications using the KbR modelling approach," in *Proceedings of ICA'17*, (Beijing, China), pp. 84–89, 2017.

[18] N. Visic and D. Karagiannis, "Developing conceptual modeling tools using a DSL," in *Proceedings of KSEM'14*, (Sibiu, Romania), pp. 162–173, 2014.

[19] H. Österle, J. Becker, U. Frank, T. Hess, D. Karagiannis, H. Krcmar, P. Loos, P. Mertens, A. Oberweis, and E. J. Sinz, "Memorandum on design-oriented information systems research," *European Journal of Information Systems*, vol. 20, no. 1, pp. 7–10, 2011.

[20] A. Hevner and S. Chatterjee, *Design Research in Information Systems: Theory and Practice*, vol. 22 of *Integrated Series in Information Systems*. New York: Springer, 2010.

[21] E. Miron, C. Muck, D. Karagiannis, and D. Götzinger, "Transforming storyboards into diagrammatic models," in *Proceedings of Diagrams'18*, (Edinburgh, UK), pp. 770–773, 2018.

[22] D. Karagiannis and H. Kühn, "Metamodelling platforms," in *Proceedings of EC-Web'02*, (Aix-en-Provence, France), pp. 182–182, 2002.

[23] D. Götzinger, E.-T. Miron, and F. Staffel, "OMiLAB: An open collaborative environment for modeling method engineering," in *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools* (D. Karagiannis, H. C. Mayr, and J. Mylopoulos, eds.), pp. 55–76, Cham: Springer, 2016.

[24] D. Karagiannis and M. Schwab, "An engineering approach for the design of hybrid modelling methods," in *Enterprise Information Systems: 14th International Conference, ICEIS 2012, Wroclaw, Poland, June 28 - July 1, 2012, Revised Selected Papers* (J. Cordeiro, L. A. Maciaszek, and J. Filipe, eds.), pp. 3–17, Berlin, Heidelberg: Springer, 2013.

[25] N. Efendioglu, R. Woitsch, and W. Utz, "A toolbox supporting agile modelling method engineering: Adoxx.org modelling method conceptualization environment," in *Proceedings of PoEM'16*, (Skövde, Sweden), pp. 317–325, 2016.

[26] J. Venable, J. Pries-Heje, and R. Baskerville, "A comprehensive framework for evaluation in design science research," in *Proceedings of DESRIST'12*, (Las Vegas, NV, USA), pp. 423–438, 2012.

[27] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI," *IEEE Internet Computing*, vol. 6, no. 2, pp. 86–93, 2002.

[28] I. Tachmazidis, S. Batsakis, J. Davies, A. Duke, M. Vallati, G. Antoniou, and S. S. Clarke, "A hypercat-enabled semantic internet of things data hub: Technical report," *CoRR*, vol. abs/1703.00391, 2017.

[29] M. E. Gregori, J. P. Cámara, and G. A. Bada, "A jabber-based multi-agent system platform," in *Proceedings of AAMAS'06*, (Hakodate, Japan), pp. 1282–1284, 2006.

[30] T. Mayerhofer, P. Langer, M. Wimmer, and G. Kappel, "xMOF: Executable DSMLs based on fUML," in *Proceedings of SLE'13*, (Indianapolis, IN, USA), pp. 56–75, 2013.

[31] G. Dévai, M. Karácsony, B. Németh, R. Kitlei, and T. Kozsik, "UML model execution via code generation," in *Proceedings of EXE@MoDELS'15*, (Ottawa, Canada), pp. 9–15, 2015.

[32] S. Wolny, "A runtime model for SysML," *Doctoral College Cyber-Physical Production Systems*, pp. 43–48, 2017.

[33] T. Morita, N. Kashiwagi, A. Yorozu, M. Walch, H. Suzuki, D. Karagiannis, and T. Yamaguchi, "Practice of multi-robot teahouse based on printeps and evaluation of service quality," in *Proceedings of COMPSAC'18*, (Tokyo, Japan), pp. 147–152, 2018.

[34] H.-G. Fill and D. Karagiannis, "On the conceptualisation of modelling methods using the ADOxx meta modelling platform," *Enterprise Modelling and Information Systems Architectures - An International Journal*, vol. 8, no. 1, pp. 4–25, 2013.